

REMARKS/ARGUMENTS

Claims 1-14 are pending in the application. Reconsideration and a withdrawal of the rejection is respectfully requested.

The rejections are based on newly cited art, namely a single U.S. Patent and that reference in combination with another. Applicant respectfully traverses the rejections for the reasons set forth herein.

Claims 1-4 and 6 stand rejected under 35 USC 102(e) as being anticipated by Nachenberg (US 6,851,057). This rejection is respectfully but strenuously traversed and reconsideration and a withdrawal of the rejection are hereby requested.

The Examiner has rejected claims 1-4 and 6 on the basis that Nachenberg is considered to disclose a virus detection system that operates under P-code to detect the presence of a virus in a file having multiple entry points (citing to the abstract). The Examiner further considers that the virus detection method includes interpreting code with an interpreter (col. 7, lines 35-40); evaluatively writing with said interpreter, the results of said interpretation (col. 7, lines 60-65); wherein the interpreter evaluates the P-code in the P-code data file and writes to the data file; and scanning the results of said interpretation for the presence of proscribed code and identifying proscribed code in said results (col. 7 line 66 to col. 8 line 5); and making the results available to a reporter for reporting (col. 3, lines 64-67).

The Examiner also considers that Nachenberg scans for the presence of code of interest (citing to col. 7 lines 45-67), and applies this to reject claim 2.

The Examiner also considers that claim 4 is anticipated because Nachenberg is considered to disclose a method wherein scanning the results of the interpretation for the

presence of proscribed code further comprises scanning for the presence of code of interest (col. 7, lines 45-67 in combination with col. 8 lines 1-5).

Applicant's present invention is not disclosed or suggested by Nachenberg. Applicant's claimed method and apparatus relate to and recite interpreting suspect code with an interpreter and evaluatively writing the results of said interpretation, and further includes the step of scanning the results of said interpretation for the presence of proscribed code. Applicant's invention is an improvement over the prior art.

The present invention is designed to combat disguised code. The preferred embodiments interpret suspect code as part of a virus scanning engine by first interpreting the code. Unlike emulators, compilers and interpreters commonly used in the prior art which execute code as it is interpreted, however, the interpreter used in the preferred embodiments do not execute code. Rather the interpreter used in the preferred embodiments first summarily evaluates code and then writes the results to a results table for further evaluation. This process of the preferred embodiments results in *detection of the results of the interpreted code*. This review of these results in the table is an improvement over the emulators, compilers and interpreters used in the prior art process *which detected the behavior of the code*.

(Specification ¶ [0021], emphasis added)

According to the Applicant's present invention, what is done to accomplish the software virus detection method involves interpreting code that is suspected of containing proscribed code with an interpreter and evaluatively writing with said interpreter the results of said interpretation. The results of the interpretation (i.e., interpreted results) are then scanned for the presence of proscribed code.

The Nachenberg reference discloses a virus detection system (VDS) 400 that includes a P-code data file 410, a virus definition file 412 and an engine 414. However, the P-code data file is disclosed to hold P-code instructions for examining the host file

100. Nachenberg, contrary to the present invention, discloses that the virus detection software (VDS) 400, includes an engine 414 with an emulating module 426. According to the Applicant's invention, the evaluative writing of the interpreted code is what is provided as the result which is scanned for the presence of proscribed code.

Nachenberg even distinguishes the Applicant's invention, where in Nachenberg, there is a disclosure that if the P-code does not directly detect a virus, then the VDS 400 performs a scan of the file. Accordingly, under Nachenberg's method the results are not what is scanned, but rather the file 100 thought to contain the suspect code. Thus, it would appear that the citation of Nachenberg relied on does not correspond to the Applicant's claimed invention where scanning, as claimed, is of the results of evaluative interpreting.

Again, though Nachenberg does briefly refer to P-code identification without emulating or string searching (col. 9, lines 1-2), there is no mention of interpreting suspect code with an interpreter and evaluatively writing the results of the interpretation, and scanning the results (as opposed to scanning the file itself). Nachenberg's disclosure provides one instance where the P-code "directly detects" a virus, and a second instance, where the P-code does not "directly detect" a virus, and the VDS 400 performs a scan of regions of the file 100. Unlike Applicant's claimed method, neither of the Nachenberg possibilities appears to disclose scanning where the scanning is of the results that the interpreter has evaluatively written.

What Nachenberg's disclosure appears to be interpreting, as the Examiner's cited passage referred to indicates, is a P-code data file (program code instructions in an interpreted computer language), and not, as Applicant claims, the suspect code (e.g., see

the host file 100 of Nachenberg). The P-code data file is described to contain P-code instructions. Nachenberg even states that the P-code instructions are created by writing instructions in a computer language and then compiling the instructions into P-code. Applicant, however, provides a method which distinguishes the prior methods (see Applicant's specification at [0021] reproduced above), such as Nachenberg, which use emulators and compilers which execute code as it is interpreted. According to the Applicant's claimed invention, the results of the evaluative interpretation of the suspect file are scanned for proscribed code, whereas Nachenberg discloses scanning the suspect file itself ("the P-code determines 512 which areas of the file 100 should be scanned for virus strings"). Accordingly, Nachenberg does not disclose the Applicant's claimed invention.

Applicant utilizes the evaluative results by scanning them, and, from the scan of the evaluative results, determines whether proscribed code is present. As Applicant states in the specification, the review of the results in the table is an improvement over the emulators, compilers and interpreters used in the prior art process *which detected the behavior of the code. (Specification [0021])*

Contrary to the Applicant's invention, emulating requests also are referred to in conjunction with the P-code are disclosed by Nachenberg, and support a further reason why Nachenberg does not disclose the present invention. Considering the Nachenberg reference for what it fairly discloses, emulation would appear to be taking place as part of its method. If emulation takes place, it would not, as in the Applicant's invention be taking place on the "results" claimed by the Applicant, as these results are interpreted results and do not appear to be able to be compiled and emulated. Rather the interpreted

results are evaluated and analyzed. Nachenberg's contrary disclosure appears at col. 9, lines 15-24:

Scan requests posted by the P-code are preferably merged and minimized to reduce redundant scanning. For instance, a posted request to scan bytes 1000 to 1500, and another posted request to scan bytes 1200 to 3000, are preferably merged into a single request to scan bytes 1000 to 3000. Any merging algorithm known to those skilled in the art can be used to merge the scan requests. Posted emulating requests having identical contexts can also be merged, although such posts occur less frequently than do overlapping scan requests.

According to the Applicant's invention, the interpreter is an evaluative interpreter that reads and evaluates code but writes the results to a table of interpreted results. The table may be a buffer or other memory. (See the Applicant's specification [0025]) Applicant's interpreter permits higher speed operation than standard executable interpreters. Applicant's method utilizes open read file call that permits opening and review of the files that may be associated with the code.

New claim 14 has been added to more clearly distinguish the Applicant's invention, and articulate these features. According to Applicant's method, the interpreter will interpret each line of the code, and as it is interpreting looks for two instructions, file open write and file open modify. A pointer may be set in the code sequence and an emergency flag set, and an emergency message sent to the reporter so that an alert may be issued. Unlike Nachenberg, the Applicant's evaluative interpreter provides interpreted results, and the interpreted results are then scanned by a pattern analyzer. Even assuming Nachenberg's P-code disclosure would be considered by the Examiner to identify 520 viruses in the file 100 without emulating or string searching, Nachenberg still fails to

disclose the Applicant's method wherein the interpreter produces evaluative interpreted results of the code and the results are then scanned.

Nachenberg provides program instructions in an interpreted computer language, and defines that as the P-code, and further states that the P-code data file 410 holds instructions for examining the host file 100. This would not appear to be interpreted results – as the Applicant's method discloses and claims. According to Nachenberg, the P-code interpreter 418 interprets P-code in the P-code data file, which Nachenberg states are the instructions for examining the host file 100. The instructions are created by writing instructions and compiling the instructions into P-code. The interpreted P-code is then used to control the operation of the engine 414.

Nachenberg does not appear to utilize the results of interpretation and then scan those results to determine the presence of proscribed code. Rather, turning to the passages in Nachenberg, the scanning module is not disclosed to scan what the Applicant's present invention scans, namely, the interpreted results, but rather virtual memory 434 or regions of a file 100 for virus signatures held in the VDF 412.

Nachenberg does not mention that the scanner scans the results of an evaluative interpretation. Applicant's invention discloses and claims interpreting the suspect file and evaluatively writing with said interpreter the results of said interpretation, and thereafter scanning *the results of said interpretation* for the presence of proscribed code and identifying proscribed code in said results, and making said results available to a reporter for reporting or to a processor for further processing of said results with a processing component.

The distinction is further supported and confirmed by considering that the scanning engine is disclosed in Nachenberg to include an emulator to emulate entry points of a virus so that the virus may become apparent. Even the Nachenberg claims specify the emulator component in conjunction with carrying out the method. Applicant's invention is distinguishable from Nachenberg. Even according to the embodiments that Nachenberg describes (see claim 8), what is referred to is executing instructions stored in an intermediate language representation. This is not the same as evaluatively writing results, it is executing. The p-code data file is disclosed in Nachenberg to contain the instructions. At the same time, it would appear that the rejection relies on Nachenberg for a disclosure of "interpreting code with an interpreter":

The engine 414 controls the operation of the VDS 400. The engine 414 preferably contains a P-code interpreter 418 for interpreting the P-code in the P-code data file 410. The interpreted P-code controls the operation of the engine 414. In alternative embodiments where the data file 410 holds instructions in a format other than P-code, the engine 414 is equipped with a module for interpreting or compiling the instructions in the relevant format. For example, if the data file 410 holds JAVA instructions, the engine 414 preferably includes a JAVA Just-in-Time compiler.

(Nachenberg, col. 7 lines 35-45)

Nachenberg would appear to be different than the Applicant's use of the interpreter to interpret the suspect file. Rather, the relied on disclosure of Nachenberg is not providing interpreting of the suspect file, as Applicant's invention does, but refers to interpreting program code instructions of a P-code data file.

The reliance on Nachenberg further appears to be misplaced and is distinguishable, in that the reference and reliance on Nachenberg, col. 7, line 66 to col. 8,

line 6, discloses scanning not interpreted results (as the Applicant's invention provides), but rather the file 100 itself. The cited portion of Nachenberg therefore does not correspond with the Applicant's method as disclosed and claimed.

NEW CLAIM 13:

Applicant has presented new claim 13 to more particularly distinguish these features over the cited art. Claim 13 is fully supported by the specification (see e.g., par. [0027]), and no new matter has been introduced.

New claim 13 provides that the interpreter includes a plurality of facilities for enabling the interpretation of a plurality of code formats and languages in source or compiled form. Nachenberg's disclosure, as applied in the rejection, would appear to be contrary to the Applicant's description of Applicant's interpreter, as Applicant's interpreter may be used in conjunction with many code formats and languages. The P-code interpreter of Nachenberg is not provided to accomplish this, but rather, would be limited. Nachenberg's disclosure appears to specifically acknowledge this, providing the engine 414, and not the p-code interpreter 418, must be equipped with a module for interpreting or compiling instructions in a relevant format. Therefore, not only is Nachenberg different for the reasons discussed above, but also, the p-code interpreter disclosure relied on in the rejection does not meet the function of the Applicant's claimed method, and what is described and claimed as Applicant's interpreter feature.

Claims 5, 7-12 stand rejected under 35 USC 103(a) as being unpatentable over Nachenberg, as applied to claims 1 and 4, and further in view of US 5,728,901 ("Shieh").

This rejection is respectfully but strenuously traversed and reconsideration and a withdrawal of the rejection are hereby requested.

First, for the reasons set forth above, Applicant's invention as recited in claims 1 and 4 is not anticipated, nor is it obvious in view of Nachenberg alone, or even when the further reference of Shieh is combined.

Second, though Shieh is relied on for a disclosure of a pattern analyzer, Shieh related to a pattern-oriented intrusion detection system and method that defines patterns of intrusion based on object privilege and information flow in secure computer systems to detect actual intrusion occurrences. (Shieh col. 2, lines 13-17), and moreover, is based on capturing the dynamic flow of object privileges and data, thereby enabling the definition and discovery of specific intrusion patterns in audit trails. (Shieh, col. 4, lines 40-44).

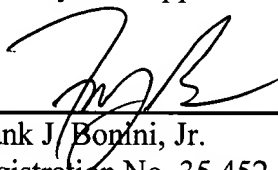
One of ordinary skill in the art would not have looked to Shieh, which relies on dynamic flow capturing for discovery of intrusion patterns in a secured computer system to combine with Nachenberg.

For the above reasons, Applicant's invention is not taught, suggested or disclosed by the combination of Nachenberg and Shieh.

Applicant's invention is distinguishable over the cited references and is not disclosed thereby. For the reasons set forth herein, Applicant respectfully requests reconsideration and a withdrawal of the rejections.

In the event an extension or further extension of time is required, one is respectfully requested, and the Commissioner is hereby authorized to charge the Applicant's undersigned representative's deposit account for any fees which may be required in connection with any extension or the filing of this amendment/response.

Respectfully submitted,
JOHN F. A. EARLEY III
FRANK J. BONINI, JR.
CHARLES L. RIDDLE
HARDING, EARLEY, FOLLMER & FRAILEY
Attorneys for Applicant



Frank J. Bonini, Jr.
Registration No. 35,452
P.O. Box 750
Valley Forge, PA 19482-0750
Telephone: (610) 935-2300

Date: 10/18/07